EE420 – Digital Image Processing

Homework 5

**Assigned: March 02, 2021 Due: March 11, 2021**

**Maximum Possible Grade: 50 points**

General guidelines

Please upload your response to Canvas as a ZIP file with the following filename convention

* Name of ZIP file FIRSTNAME\_LASTNAME\_Homework6.zip
* Replace FIRSTNAME with your first name
* Replace LAST NAME with your last name

**Failure to adhere to the filename convention will result in deduction of points.**

Provide a detailed response to each question, including supporting mathematical arguments. Include screenshots of image before and after image processing. Failure to do will result in deduction of points.

Submitting your solutions

Please ensure that the ZIP file uploaded to Canvas, includes the following components:

* PDF file of your write-up including response to all questions.
* Completed MATLAB Source code for hough\_Transform\_for\_Circles.m
* Completed MATLAB Source code for HWK6\_DetectCircles.m

**Failure to adhere to the filename convention will result in deduction of points.**

Objective

As part of this assignment you will be implementing an algorithm to detect circles of known radius, in an image, using the Hough Transform discussed in the lecture. Your task is to identify as circles in the MATLAB image **coloredChips.png**.

### Brief description

The key steps in implementing the Hough transform are enumerated below:

* identify pixels to vote
* accumulate votes
* detect peaks and
* identify associated circles.

For the purpose of this assignment we will assume that the radius of the circle is known. Your task is to find circles with the specified radius in the image. As an example, I can search for all circles with radius 28.5 pixels in the coins image shipped with MATLAB. A visual representation of key steps in the workflow are furnished in Figure 1.

The radius of 28.5 pixels was identified first by visual inspection of the coins image and subsequently refined by examining the height of peaks in the Hough parameter space. I was looking for the tallest peaks.

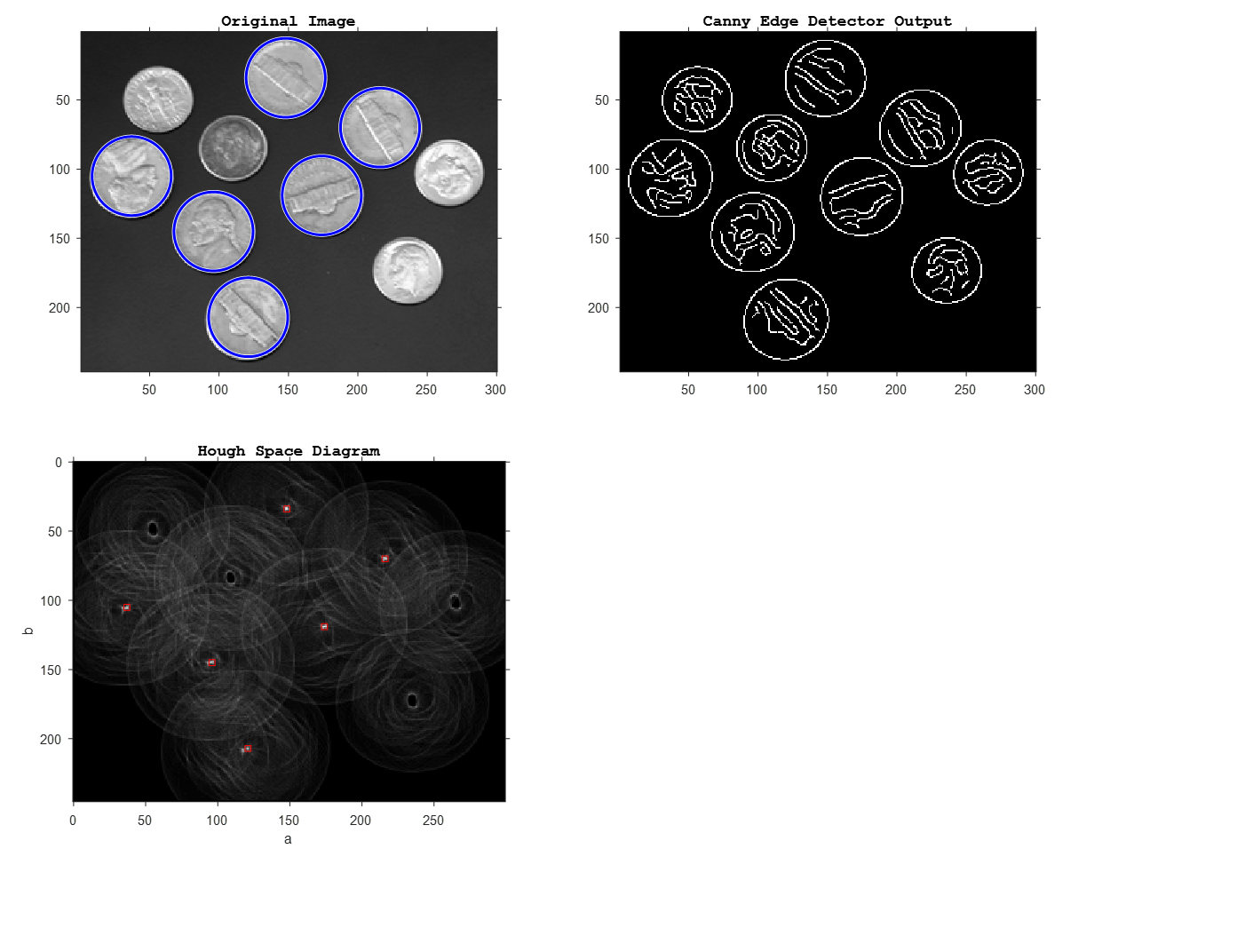


Figure 1. Detecting circles of known radius using the Hough Transform

The remainder of this document provides a detailed description of the steps involved in implementing the Hough Transform for detecting circles of a prescribed radius.

#### 1. Identify pixels to vote (NEED TO IMPLEMENT)

The first step is to identify pixels that will cast votes. This could be done by thresholding the gradient image, so that pixels whose gradient magnitude exceeds the threshold will be allowed to cast votes. The preferred approach is to use the Canny edge detector (**edge()** in MATLAB) for this task. Experiment with the parameters of the Canny edge detector to make sure you can detect all circles of interest.

#### 2. Accumulate votes (NEED TO IMPLEMENT)

Each edge pixel in the image space votes for all possible circles (with prescribed radius and unknown center ) which pass through the point . The coordinate convention is illustrated in Figure 2. The expression for disclosed in Figure 2 is consistent with the definition of a circle with radius , and center , in polar coordinates.

*Selecting :* The angle spans the interval . The number of angles examined may be restricted to , corresponding to a step size (increments) of 1-degree.

*Selecting the circle centers :* The range of values assumed by the center of each circle depends on the size of the image. The smallest/largest value of are where is the number of columns in the image. Likewise, the smallest/largest value of are where is the number of rows in the image. For ease of implementation, define as arrays with 1-pixel increments spanning the range and respectively.

The Hough space for circle fitting is a discretized grid of values. This restricts the size of the Hough Space accumulator, allowing it to be interpreted as a 2D matrix and viewed as an image. The abscissa of the Hough space (analogous to the x-axis in Cartesian coordinates) is , while the ordinate is (analogous to y-axis in Cartesian coordinates) is the parameter .

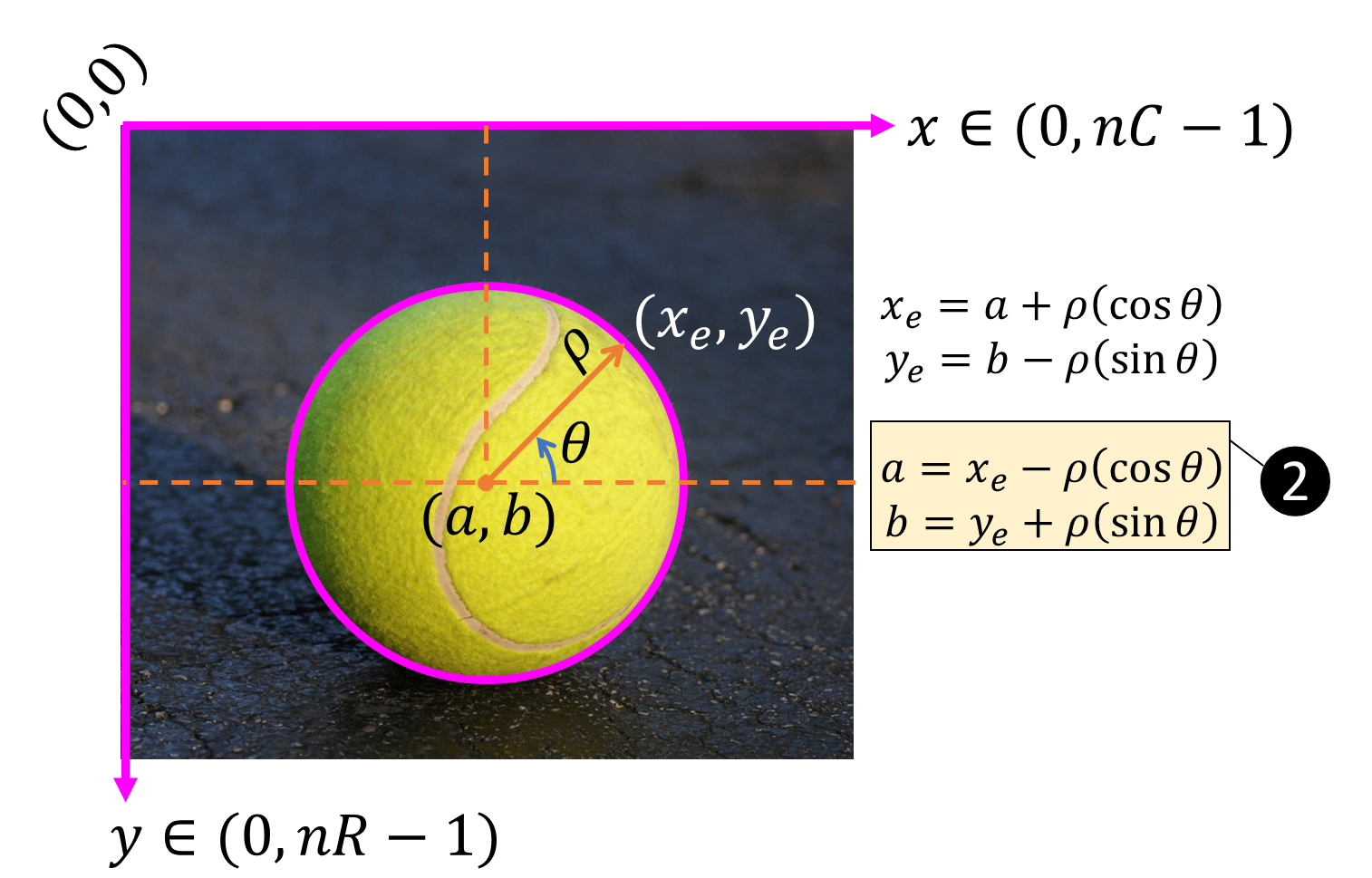


Figure 2 Coordinate Convention for detecting circles using the Hough Transform

*Accumulating votes:* Given an edge point with coordinates identify the value in Hough space that will be incremented, using Eq.(2) in Figure 2. Please bear in mind that the values of identified may not be an exact integer. You will have to search for the closest integer values in the discretized Hough parameter space. Slide-54 of Lecture-12 provides clues to implementing this in MATLAB. Please note that the procedure outlined in Slide-54 is specific to line fitting and will need to be adapted for circle fitting.

Follow the procedure outlined in Slide-72 of Lecture-12 to accumulate votes for each edge pixel. Please disregard the loop over radius values.

WARNING: When implementing the Hough transform, you will likely find that the first, last row and the first, last columns of the accumulator contain spurious peaks which will affect the peak detection process. The problem may be addressed by deliberately zeroing out the first, last row and the first, last columns of the accumulator array.

### *Detecting peaks in the Hough Transform*

As discussed in the lecture, local maxima in Hough Space correspond to circles with the prescribed radius in the image. Rather than writing code for detecting peaks (local maxima) in Hough space, we will rely on MATLAB’s built-in function houghpeaks(). The function returns the location of the row, column indices of the largest peaks in the Hough space image, where is user defined. These can be converted into values by subtracting 1 from the column, row indices respectively.

### *Overlaying circles on image (NEED TO IMPLEMENT)*

The values associated with the largest peaks in Hough Space specify the center of the circles in image space with the prescribed radius . Your final task is to overlay these circles on the displayed image. The MATLAB command viscircles(…) may be used to overlay circles as shown in Figure 1. The command accepts the center and radius of the circles to be displayed.

### Implementation Caveats

* To use the function houghpeaks, it is important that your implementation of the Hough accumulator conform to MATLAB’s convention.
* The origin of the image coordinates is at the top left corner of the image, consistent with MATLAB’s coordinate convention used in implementing the Hough transform (type help hough in MATLAB). Remember to subtract 1 from the row, column indices of the edge pixel to obtain the coordinates of an edge pixel.

### MATLAB Code to implement

The starter code HWK6\_DetectCircles.m is intended to help you get started with the assignment. It includes code for reading an input image, and makes calls to the following functions: hough\_Transform\_for\_Circles(), houghpeaks().

* Implement a MATLAB function hough\_Transform\_for\_Circles with the following interface

[HS,a,b] = hough\_Transform\_for\_Circles(bwEdgeMap)

The function accepts an input binary image bwEdgeMap. The function should return the Hough accumulator matrix HS, the array of centers a,b.

* The starter code makes a call to the built-in MATLAB function houghpeaks to help you identify the peaks in Hough space. The function returns a matrix P that contains the row, column indices of the largest peaks in Hough Space. You are free to choose the value N. If N=6, the matrix P returned by houghpeaks will have 6 rows and 2 columns, provided it could find 6 peaks in Hough Space. You may have to play with the Threshold parameter to detect all circles in the image. Pick the threshold to be a percentage of the maximum value in the Hough accumulator image.

**Deliverables & Questions**

1. MATLAB code for houghTransform\_for\_Circles.m and HWK6\_DetectCircles.m (35 points)
2. Explain why the expressions for listed in Eq.(2) have different sign for and ?

(2 points)

1. What is your estimate of the radius of the colored chips? What were your parameters for the Canny edge detector? Please justify your choice. (1 points)
2. Were you able to detect the 28 circles in the image? If not comment on why you are unable to detect all the circles. (1 points)
3. What happens to the number of detected circles when you pick a large value for the threshold in houghpeaks? (2 points)
4. What happens to the number of detected circles when you pick a small value for the threshold in houghpeaks? (2 points)
5. Modify the code so that you only detect the yellow chips in the image **coloredChips.png**. (HINT: Check the RGB values of the center of each circle). (2 points)
6. Screenshots of the original image, output of Canny edge detector, the Hough Space diagram with the peaks overlaid. Label each screenshot clearly. Failure to do so will result in deduction of points. (5 points)